ECE-6483 Real Time Embedded Systems
Midterm 2

New York University, Spring 2025

Due on 04/30, 11:59 PM

Name: Raman Kumar Jha
NYU ID: N13866145

During the COVID pandemic, one of the most monitored biometric parameters measured was blood oxygen saturation, or SpO2. One of the most available sensors to make this measurement, as well as heart rate is the MAX30100 chip. Attached is the datasheet for your review.

Review the I2C section of the datasheet.

Question 1

The objective of this question is to connect to the MAX30100 chip and retrieve SPO2 information (Red LED and IR LED) from the FIFO buffer. We also wish to capture the interrupt and toggle the onboard LED every time a measurement is received.

1. Read the relevant parts of attached datasheet to familiarize yourself to the device. Specifically focus on the serial communications part (pages 23-26), the register map shown on page 10.
2. Using the pin diagram on Page 8, sketch a schematic that connects your STM development board's I2C bus (review the STM datasheet for SCL/SDA pin information) to the correct pins on the MAX30100. Don't forget pull up resistors. Also connect the INT pin from the sensor to a generic GPIO pin (you can choose which one). Pick wisely though since you will need to write an interrupt handler for this pin to capture when an SP02 measurement is ready.
3. Start a new project in PlatformIO, chose your STM board, and select MBED framework.
4. In your void main() , Write the code to:
   a. Setup the `DigitalOut` for the onboard LED.
   b. Configure the sensor:
      i. Using the I2C object (i.e. `i2c.transfer()`), Configure register address `0x01` to enable only the SP02 interrupt.
      ii. Set sensor mode to SP02 (See register address `0x06`, MODE BITS)
      iii. Leave default settings in SP02 Config register (`0x07`)
   c. Using MBED `InterruptIn`, set up the pin on your microcontroller to receive the INT from the sensor when an SP02 measurement is ready.

5. Write a function called GetData() that communicates I2C to the chip and retrieves (and returns) the values of the sensor's Red LED and the IR LED. This involves:
    i. Reading the FIFO Data register (0x05). Be sure to fully read the DATASHEET pages 12-15 to understand fully how the FIFO works and how the data for the 2 sensor LEDs is provided. Pseudo code is provided in the datasheet as well. You can use any of the I2C functions provided in the MBED I2C API.
6. Write the ISR for the function specified in InterruptIn that you configured to receive the SP02 measurement RDY interrupt. You can call GetData() from 5. Also write the code to display the SHORT value of both the Red LED and the IR LED to the terminal. You should also blink the on board LED each time the interrupt is received.
7. Submit 2 files, your schematic and your code file with full comments.

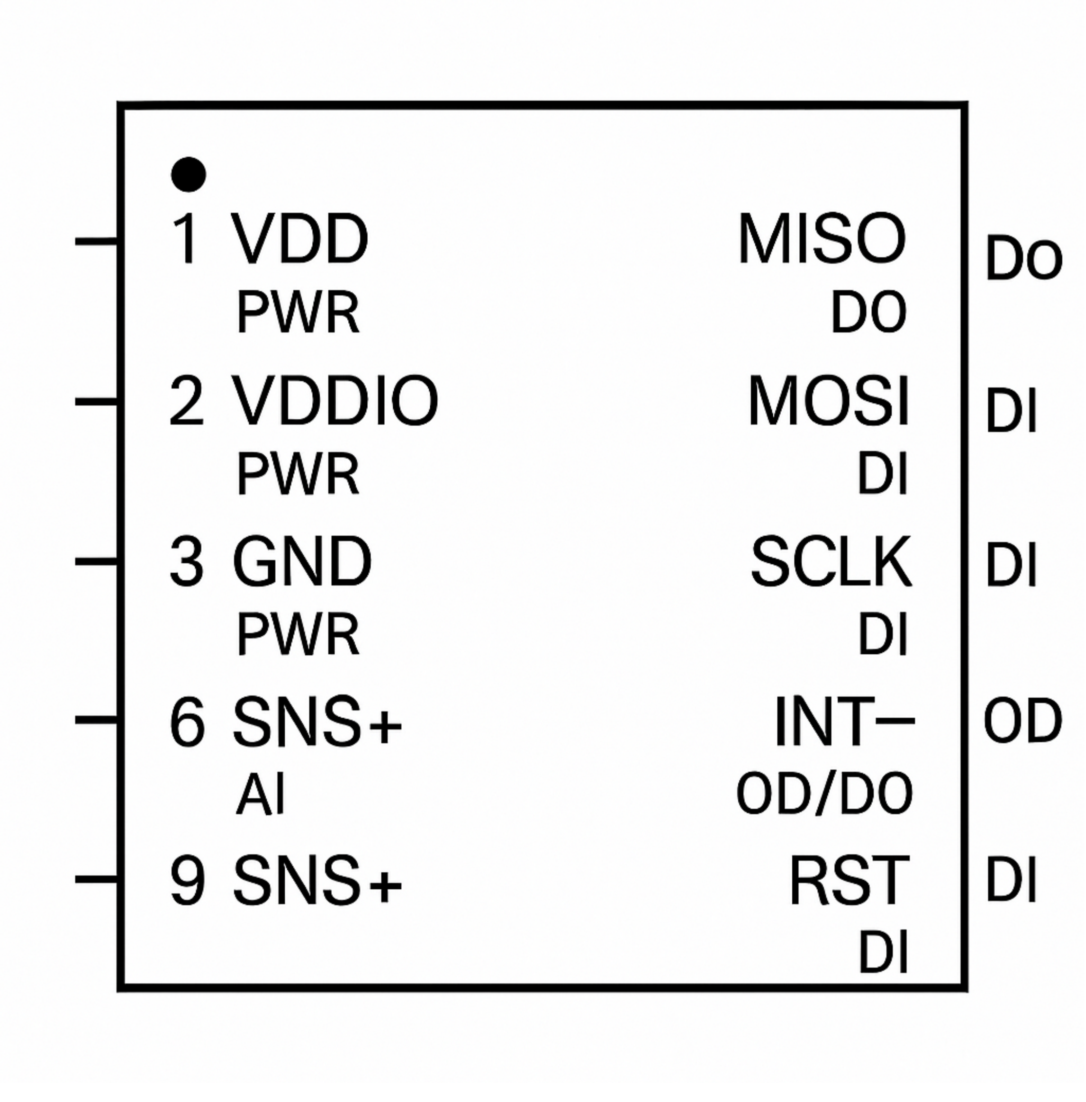
    Solution: Both of the files, i.e., schematic and the code file named midterm2_q1.cpp, are attached.

Question 2

Suppose you were tasked with designing your own device that measures respiratory rate, to be used along side the MAX30100 described above that used SPI for communication. Please read the whole question before beginning your design.

1. Propose a pinout diagram for your device that provides the necessary power pins, GPIO pins, and communications pins.

Solution:



2. What interrupts would your design provide, if any, that the remote microcontroller may interface to?

Solution:
INT is low when any enabled flag is set in INT_STATUS ($0x00$). Flags are cleared on read of INT_STATUS.

Flag	Bit	INT Trigger	Clear Mechanism	Purpose / Notes
RR_RDY	5	Yes	Read INT_STATUS	Indicates a new respiratory-r sample (16-bit RR + 8-bit qu ity code) is available in the FII Primary interrupt used for re time monitoring.
TEMP_RDY	6	Yes	Read INT_STATUS	Temperature conversion is co plete; result available in TEMP_I and TEMP_FRAC. Optional for th mal compensation.
FIFO_ALM	4	Yes	Read INT_STATUS and drain FIFO	Triggered when FIFO level threshold in ALM_CFG (0x08). U ful in burst or low-power sa pling modes.
FAULT	2	Yes	Read INT_STATUS	Triggered by sensor malfuncti overflow, or electrical fault. H may reset via MODE_CFG.

Table 1: RR200 interrupt summary. INT is asserted (low) whenever any enabled flag is high. All flags are cleared by reading INT_STATUS (0x00).

3. Propose a register mapping, similar to that of the MAX30100 that provides an external microcontroller the capabilities to configure your device, interrupts, etc. and retrieve the relevant information you choose to provide. Be sure to specify default values and bit specifications of each memory location.

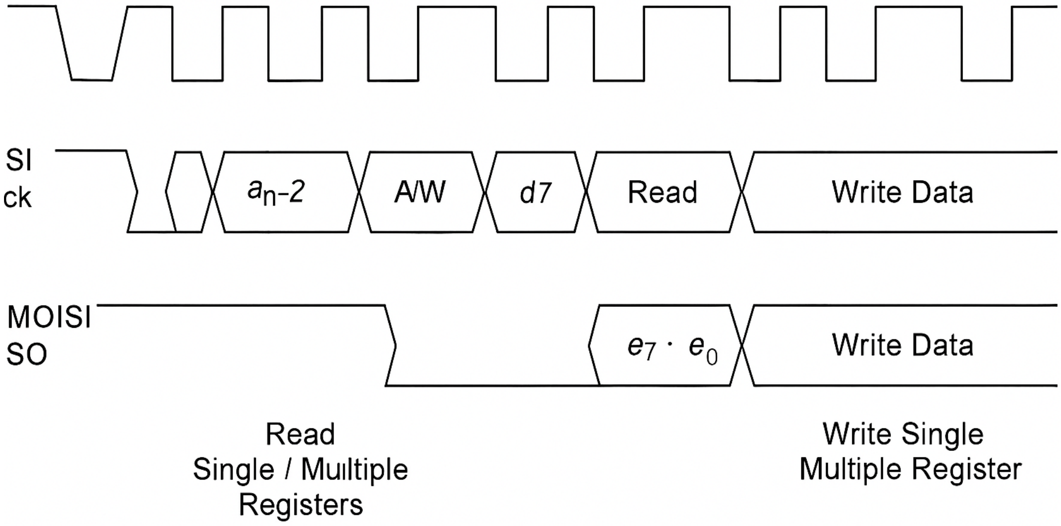
Solution:

Addr	Name	R/W	Reset	Bit 7 . . . 0 description
0x00	INT_STATUS	R	0x00	[6] TEMP_RDY, [5] RR_RDY, [4] FIFO_ALM, [2] FAULT (others 0)
0x01	INT_ENABLE	R/W	0x00	Same bit positions as INT_STATUS; 1 = enable flag in logic-OR to INT_STATUS.
0x02	FIFO_WR_PTR	R/W	0x00	Write-pointer (0–31).
0x03	OVR_CNT	R	0x00	FIFO overflow counter (read resets).
0x04	FIFO_RD_PTR	R/W	0x00	Read-pointer (0–31).
0x05	FIFO_DATA	R	—	Stream of 3-byte samples: RR_H, RR_L, QUAL. Autoincrements pointers.
0x06	MODE_CFG	R/W	0x00	[7] RESET (1 → soft-reset), [2:0] MODE{000=STBY, 001=RR+TEMP, 010=RR_ONLY, 011=TEMP_ONLY}.
0x07	RR_CFG	R/W	0x23	[7] HIRES_EN, [6:4] SR_SEL{000=10sps. . . 101=100sps}, [3:0] PWIDTH{0000=100 μs . . . 1111=1600 μs}.
0x08	ALM_CFG	R/W	0x1E	[7:4] FIFO_ALMOST_FULL level (default 30), lower nibble reserved.
0x09	TEMP_INT	R	—	Signed integer part of temperature (°C).
0x0A	TEMP_FRAC	R	—	Fractional part (LSB = 0.0625 °C).
0x0B	REV_ID	R	0x13	Silicon revision code.
0x0C	PART_ID	R	0x20	Fixed device ID (0x20).

Table 2: RR200 register summary; unused bits read 0 and are ignored on write.

4. Show a sample SPI communications timing diagram that indicates how to communicate (read/write single/multiple registers) to your device.

Solution:



Sample SPI Communications Timing Diagram

5. Write a sample code segment that you would include in your datasheet that would help the end user Interact with your device in a typical use case.

Solution: The code file is attached with the name midterm2_q2.cpp.